# Defense in Depth

Sean Barnum, Cigital, Inc. [vita[3]]
Michael Gegick, Cigital, Inc. [vita[4]]



2005-09-13 L4 / D/P, L[5]

Layering security defenses in an application can reduce the chance of a successful attack. Incorporating redundant security mechanisms requires an attacker to circumvent each mechanism to gain access to a digital asset. For example, a software system with authentication checks may prevent an attacker that has subverted a firewall. Defending an application with multiple layers can prevent a single point of failure that compromises the security of the application.

## Detailed Description Excerpts

According to Viega and McGraw [Viega 02] in Chapter 5, "Guiding Principles for Software Security," in "Principle 2: Practice Defense in Depth" from pages 96-97:[9]

> The idea behind defense in depth is to manage risk with diverse defensive strategies, so that if one layer of defense turns out to be inadequate, another layer of defense will hopefully prevent a full breach. This principle is well known, even beyond the security community; for example, it is a famous principle for programming language design: Defense in Depth: Have a series of defenses so that if an error isn't caught by one, it will probably be caught by another.[10]
>
> Let's go back to our example of bank security. Why is the typical bank more secure than the typical convenience store? Because there are many redundant security measures protecting the bank, and the more measures there are, the more secure the place is.
>
> Security cameras alone are a deterrent for some. But if people don't care about the cameras, then a security guard is there to physically defend the bank with a gun. Two security guards provide even more protection. But if both security guards get shot by masked bandits, then at least there's still a wall of bulletproof glass and electronically locked doors to protect the tellers from the robbers. Of course if the robbers happen to kick in the doors, or guess the code for the door, at least they can only get at the teller registers, since we have a vault protecting the really valuable stuff. Hopefully, the vault is protected by several locks, and cannot be opened without two individuals who are rarely at the bank at the same time. And as for the teller registers, they can be protected by having dye-emitting bills stored at the bottom, for distribution during a robbery.
>
> Of course, having all these security measures does not ensure that our bank will never be successfully robbed. Bank robberies do happen, even at banks with this much security. Nonetheless, it's pretty obvious that the sum total of all these defenses results in a far more effective security system than any one defense alone would.
>
> The defense in depth principle may seem somewhat contradictory to the "secure the weakest link" principle, since we are essentially saying that defenses taken as a whole can be stronger than the weakest link. However, there is no contradiction; the principle "secure the weakest link" applies when components have security functionality that does not overlap. But when it comes to redundant security measures, it is indeed possible that the sum protection offered is far greater than the protection offered by any single component.

3. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)
4. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/345-BSI.html (Gegick, Michael)
9. All rights reserved. It is reprinted with permission from Addison-Wesley Professional.
10. MacLennan, Bruce. *Principles of Programming Languages*. Holt,Rinehart and Winston, 1987.

A good real-world example where defense-in-depth can be useful, but is rarely applied, is in the protection of data that travel between various server components in enterprise systems. Most companies will throw up a corporate-wide firewall to keep intruders out. Then they'll assume that the firewall is good enough, and let their application server talk to their database in the clear. Assuming that the data in question are important, what happens if an attacker manages to penetrate the firewall? If the data are also encrypted, then the attacker won't be able to get at them without breaking the encryption, or (more likely) breaking onto one of the servers that stores the data in an unencrypted form. If we throw up another firewall, just around the application this time, then we can protect ourselves from people who can get inside the corporate firewall. Now they'd have to find a flaw in some service that our application's sub-network explicitly exposes, something we're in a good position to control.

Defense in depth is especially powerful when each layer works in concert with the others.

According to Howard and LeBlanc [Howard 02[11]] in Chapter 3, "Security Principles to Live By," in "Use Defense in Depth," from pages 59-60:

Defense in depth is a straightforward principle: imagine your application is the last component standing and every defensive mechanism protecting you has been destroyed. Now you must protect yourself. For example, if you expect a firewall to protect you, build the system as though the firewall has been compromised.

Unfortunately, a great deal of software is designed and written in a way that leads to total compromise when a firewall is breached. This is not good enough today. Just because some defensive mechanism has been compromised doesn't give you the right to concede defeat. This is the essence of defense in depth: at some stage you have to defend yourself. Don't rely on other systems to protect you. Put up a fight because software fails, hardware fails, and people fail. People build software, people are flawed, and therefore software is flawed. You must assume that errors will occur that will lead to security vulnerabilities. That means the single layer of defense in front of you will probably be compromised, so what are your plans if it is defeated? Defense in depth helps reduce the likelihood of a single point of failure in the system.

Important: Always be prepared to defend your application from attack because the security features defending it might be annihilated. Never give up.

**Example**

Let's quickly revisit the castle example from the first chapter. This time, your users are the noble family of a castle in the 1500s, and you are the captain of the army. The bad guys are coming, and you run to the lord of the castle to inform him of the encroaching army and of your faith in your archers, the castle walls, and the castle's moat. The lord is pleased. Two hours later you ask for an audience with the lord and inform him that the marauders have broken the defenses and are inside the outer wall. He asks how you plan to further defend the castle. You answer that you plan to surrender because the bad guys are inside the castle walls. A response like yours doesn't get you far in the armed forces. You don't give up--you keep fighting until all is lost or you're told to stop fighting.

Here's another example, one that's a little more modern. Take a look at a bank. When was the last time you entered a bank to see a bank teller sitting on the floor in a huge room next to a massive pile of money. Never! To get to the big money in a bank requires that you get to the bank vault, which requires that you go through multiple layers of defense. Here are some examples of the defensive layers:

- There is often a guard at the bank's entrance.
- Some banks have time-release doors. As you enter the bank, you walk into a bulletproof glass capsule. The door you entered closes, and after a few seconds the glass door to the

11. #dsy347-BSI_refs

bank opens. This means you cannot rush in and rush out. In fact, a teller can lock the doors remotely, trapping a thief as he attempts to exit.

- There are guards inside the bank.
- Numerous closed-circuit cameras monitor the movements of every one in every corner of the bank.
- Tellers do not have access to the vault. (This is an example of least privilege, which is covered next.)
- The vault itself has multiple layers of defense, such as:
  - It opens only at certain controlled times.
  - It's made of very thick metal.
  - Multiple compartments in the vault require other access means.

According to NIST [NIST 01] in Section 3.3, "IT Security Principles," from page 9:

> Implement layered security (ensure no single point of vulnerability). Security designs should consider a layered approach to address or protect against a specific threat or to reduce a vulnerability. For example, the use of a packet-filtering router in conjunction with an application gateway and an intrusion detection system combine to increase the work-factor an attacker must expend to successfully attack the system. Adding good password controls and adequate user training improves the system's security posture even more.
>
> The need for layered protections is especially important when commercial-off-the-shelf (COTS) products are used. Practical experience has shown that the current state-of-the-art for security quality in COTS products does not provide a high degree of protection against sophisticated attacks. It is possible to help mitigate this situation by placing several controls in series, requiring additional work by attackers to accomplish their goals.

According to Schneier [Schneier 00] in "Security Processes":

> Provide Defense in Depth.
>
> Don't rely on single solutions. Use multiple complementary security products, so that a failure in one does not mean total insecurity. This might mean a firewall, an intrusion detection system and strong authentication on important servers.

### References


| | |
|---|---|
| [Howard 02] | Howard, Michael & LeBlanc, David. *Writing Secure Code, 2nd ed*. Redmond, WA: Microsoft Press, 2002. |
| [NIST 01] | *Engineering Principles for Information Technology Security*. Special Publication 800-27. US Department of Commerce, National Institute of Standards and Technology, 2001. |
| [Schneier 00] | Schneier, Bruce. "The Process of Security[13]." *Information Security Magazine*, April, 2000. |
| [Viega 02] | Viega, John & McGraw, Gary. *Building Secure Software: How to Avoid Security Problems the Right Way*. Boston, MA: Addison-Wesley, 2002. |


## Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com